Case-Based Reasoning as a Tool to Improve the Usability of Numerical Models

Fei Ling Woon[1], Brian Knight[2], Miltos Petridis[2]

[1] Tunku Abdul Rahman College, School of Arts and Science, Kuala Lumpur, Malaysia
f.woon@gre.ac.uk
[2] University of Greenwich, School of Computing and Mathematical Sciences,
London SE10 9LS, UK
{b.knight, m.petridis}@gre.ac.uk

Abstract. There may be advantages to be gained by combining Case-Based Reasoning (CBR) techniques with numerical models. In this paper we consider how CBR can be used as a flexible query engine to improve the usability of numerical models. Particularly they can help to solve inverse and mixed problems, and to solve constraint problems. We discuss this idea with reference to the illustrative example of a pneumatic conveyor. We describe a model of the problem of particle degradation in such a conveyor, and the problems faced by design engineers. The solution of these problems requires a system that allows iterative sharing of control between user, CBR system, and numerical model. This multi-initiative interaction is illustrated for the pneumatic conveyor by means of Unified Modeling Language (UML) collaboration and sequence diagrams. We show approaches to the solution of these problems via a CBR tool.

1 Introduction

Numerical models of physical processes can provide useful advice to engineers in many fields. However, they are often designed to simulate the evolution of systems over time, and operate in a forward time direction. Generally, the engineer will specify inputs $\underline{\mathbf{I}} =(I_1,..I_k)$, and the model will calculate output $\underline{\mathbf{O}} = (O_1,..O_l)$, where $\underline{\mathbf{O}}$ is a function of $\underline{\mathbf{I}}$.

However, the engineering problems often require a model that can be queried in an inverse fashion. A designer may want to know what inputs will give given outputs. In addition, engineers often want to add constraints to outputs, searching for the right inputs. To solve these inverse problems and constraint problems directly will require a different computational model, often difficult or impossible to construct. To solve such inverse or constraint-based problems without solving the problems directly, engineers often resort to an iterative search method: running the model, looking at the results and changing the inputs accordingly for another run. In effect, the engineer is judiciously generating cases from the numerical model.

In contrast, a database model of the process may be represented by a set of stored predicates:

$$P\ (I_1,...I_k, O_1...O_l)$$

Such a model can be queried quite flexibly using SQL, specifying either inputs or outputs, and constraints. However, such a model also suffers from some disadvantages:

- It can be a very large database, particularly if *k* and *l* are large, or if high accuracy is required.
- Queries using SQL can often give null results if the database is kept small.

The idea under examination in this paper is to use a CBR system generated using a numerical model as a flexible query engine for engineers. To illustrate some of the ideas involved, we propose to use the example of a pneumatic conveyor model. The CBR system will allow an iterative approach to problem solving, where the engineer and the system can collaborate in searching for a solution.

2 Illustrative Example: The Pneumatic Conveyor Problem

Pneumatic conveying is an important transportation technology in conveying solid bulks in industry. Attrition of powders and granules during pneumatic conveying is a problem that has existed for a long time. One of the major industry concerns is to investigate how parameters such as air velocity, loading ratio, the angle of the bend and etc. affect degradation. Such knowledge is of great use in the design of conveyors.

According to Kalman (2000), the bend in a pneumatic conveying pipeline is one of the major critical devices that contribute a major part of the pressure drop (energy consumption) and consequently causes great damage to the particles. More in-depth studies can be found in Hilbert (1984) who has studied different bend structures. Marcus *et al.* (1985) have investigated the pressure loss of different bends. Agarwal *et al.* (1985) considered acceleration length due to bends and the effects of phase density, and etc. Weinberger and Shu (1986) examined the effects of the curvature radius of a bend on the transition velocity (the gas velocity at which minimum pressure drop occurs). Finally, Bell *et al.* in their studies (1996) discovered that air velocity has the prime effect on the attrition rate.

However, in general, Kalman states in his paper (2000) that parameters affecting the attrition rate can be divided into three categories:

- The particle strength: particle material, size and shape.
- The operation parameters: particle velocity and particle concentration – loading ratio.
- The pipeline and bend structure: radius of curvature, construction material, type of bend, number of bends.

In most cases, as according to the expert, pneumatic conveyor engineers are more concerned with the set of input parameters (i.e., as mentioned above) that will produce a desirable size distribution of particles.

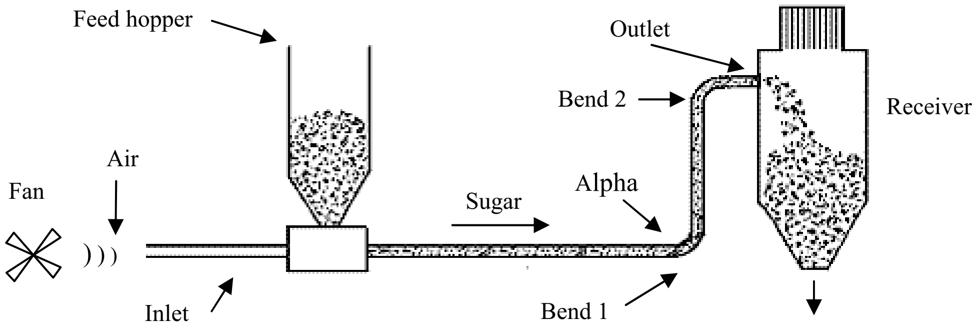
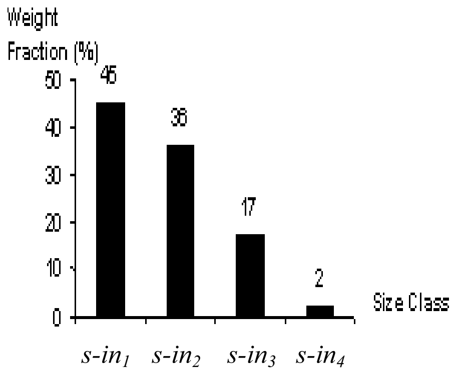


Fig. 1. The schematic diagram of a sample pneumatic conveyor



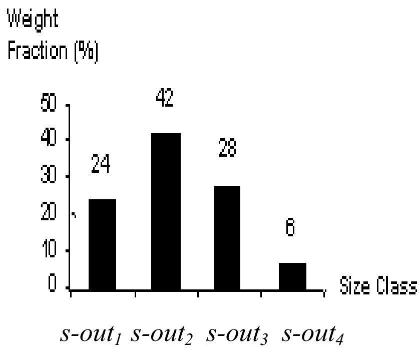


Fig. 2. Particle size distribution at the inlet **Fig. 3.** Particle size distribution at the outlet

3 Example Problems

From a designer's point of view the problem is this: we want to decide on the right angle of bend, diameter of pipe, and air velocity for a given particulate, e.g., sugar or tea. We have only 3 angles of bend and three pipe diameters available. The sugar must not degrade so that there is too much dust formed (i.e., very small particles). There may be only low power fans available sometimes, so the air velocity may be constrained. What is the best set-up to use?

We can formulate this problem as follows, representing the model in the form of a predicate, where *alpha* is the angle of the bend, *d* the diameter, *Vair* is the air velocity, $s\text{-}in_1$, $s\text{-}in_2$, $s\text{-}in_3$, $s\text{-}in_4$ are size distributions going in, and $s\text{-}out_1$, $s\text{-}out_2$, $s\text{-}out_3$, $s\text{-}out_4$ are size distributions coming out of the conveyor (i.e., Fig.1, Fig.2 and Fig.3).

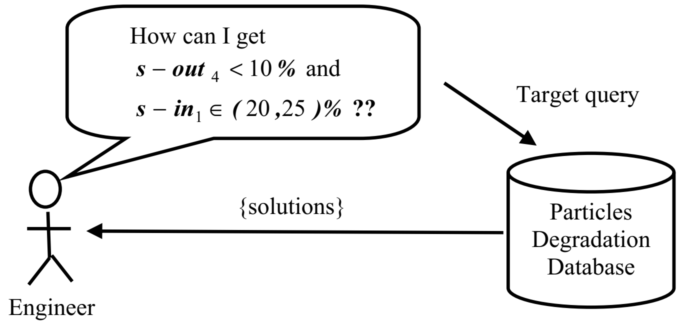


Fig. 4. An engineer makes a query to the Particles Degradation Database

With this notation, the designer's problem can be posed as a query (Fig.4):

P(?alpha, ?d, ?Vair, s-in$_1$ *, s-in*$_2$*, s-in*$_3$*, s-in*$_4$*; s-out*$_1$ *, s-out*$_2$*, s-out*$_3$*, s-out*$_4$),

?alpha $\in$ *{ 30, 45, 50},*

s-out$_4$< 10 %

s- in$_1$< 25 %

s-in$_1$>20

As a SQL query this is (for example):

Select *alpha, d, Vair* FROM P,
Where *s-in*$_1$<25 AND *s-in*$_1$>20 AND …
AND (*alpha = 30 OR alpha = 45 OR alpha = 50*)
AND *s-out*$_4$< 10;

Given that we have a large enough database, this should give a range of possible solutions. However, there are a number of problems associated with the database method. These include the following:

- The database may be expensive to produce.
- High dimensionality may make the database solution infeasible. In this example there are 7 degrees of freedom, and in order to cover the domain in reasonable detail, we might need, say, 10 points in each dimension, giving 7^{10} records altogether.
- The solution set could be very large and hence unhelpful in decision making.
- Equality constraints might mean there are no solutions at all to the SQL query.

Some of these problems can be approached by means of a CBR model. The problem of database size may be reduced somewhat by means of a sparse database of important cases. Although at first sight the dimensional catastrophe is still present, there is still the possibility that we can produce a relatively small efficient case base to replace a large database. One reason for suspecting this is that we are looking at the domain of numerical models. In this field, there is a great deal of regularity in the model, and we would expect fine detail to be well represented by some such adaptive process such as interpolation. This should allow great reduction in storage. Interpolative CBR systems have been studied by Chatterjee and Campbell (1993), and by Knight and Woon (2003), who propose a generalisation of Shepard's method known as GSNN.

Also, the other problems associated with the simple database model may also be alleviated in the CBR approach. CBR retrieval is on the whole more amenable to usability questions than is SQL, giving cases ordered by closeness to input criteria. It will always give answers, and they might be ordered according to user needs.

3.1 Mixed-Initiative Problem Solver

Having discussed what a possible designer's problem looks like, how do we now put together the expertise of the user, the CBR system, and the numerical model to solve the designer's problem? Fig. 5 shows the UML collaboration diagram (Schach, 1999) of the mixed-initiative problem solver.

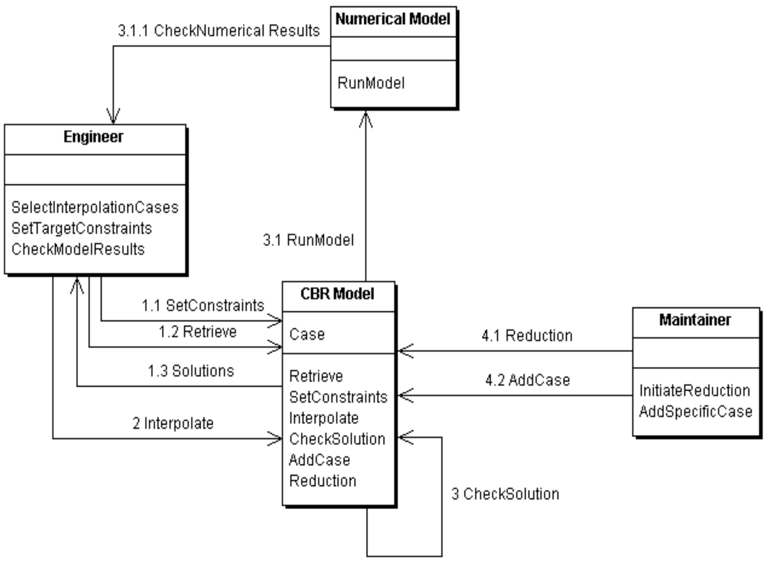


Fig. 5. The UML collaboration diagram for the mixed-initiative problem solver

As revealed in Fig. 5, there are four agents in the problem solver (i.e., the user – engineer, the CBR system – CBR model, the maintainer, and the numerical model). Each of them has its own expertise and specializes in a set of tasks. Having said that, they are also able to work together and assist one another to achieve a specific goal when necessary. The UML collaboration diagram captures the essence of the collaboration and the communication between multiple agents in achieving a set of goals. To better illustrate how it works, we provide a simple scenario (Fig. 6) of the agents solving a problem. The form of this figure is adapted from that suggested by Bridge (2002). It corresponds to the UML collaboration diagram based upon Fig. 5. Alternatively, Fig. 7 shows this scenario as a UML sequence diagram.

Action	Agent	Status
1.1 SetConstraints: User inputs constraints	CBR model	Idle
1.2 Retrieve: The user informs the CBR system to search for the nearest case. The CBR system returns a set of nearest cases and prompts for further instruction.	CBR model	Start searching
2 Interpolate: The CBR system prompts the user to select cases for interpolation. The user selects cases and informs the CBR system to ask for interpolated values.	CBR model	Compute interpolated values
3 CheckSolution: The user informs the CBR system to check results.	CBR model	Idle
3.1 RunModel: The CBR system decides the boundary of testing and informs the numerical model to run model	Numerical model	Run test
3.1.1 CheckNumerical Results: The numerical Model returns the model results and inform the user of invalid cases and valid cases	user	Analyse results

Fig. 6. The scenario of the user querying a set of constraints

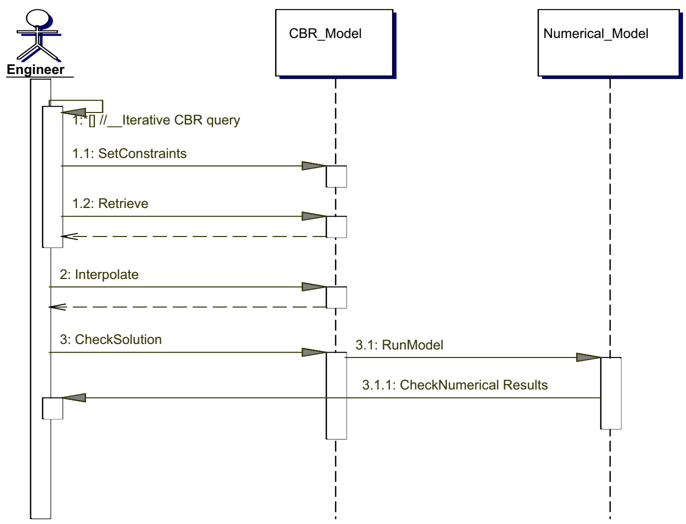


Fig. 7. The UML sequence diagram for the mixed-initiative scenario

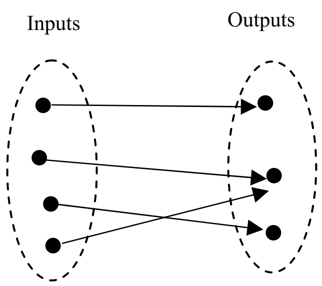
The above is merely a typical scenario of the way the mixed-initiative problem-solver might behave. In fact there are also other interesting aspects to be investigated, such as automated adaptation on cases and the maintenance of the case base.

4 The CBR Approach

We now turn to the problems to be solved with a case-based approach. In such an approach, we want to preserve the convenient property of databases that they can be queried in a flexible manner, with no distinction between inputs and outputs. Thus the inverse problem becomes just another query. In this case, the similarity metric must be defined over the whole problem and solution space. In addition, we would like to use an interpolative scheme, so as to utilise as sparse a case base as is possible. With these ideas in mind, there are a number of problems to be dealt with, as discussed below:

4.1 Multi-valued case mappings

Although in the numerical model **O** is given as a single valued function of **I**, the inverse problem cannot be assumed as single valued. As in Fig. 8, there may be several solutions to a given query where outputs are specified. If we are using 1-NN retrieval (i.e., k-Nearest Neighbour (k-NN) where k=1, (Cover and Hart, 1967)), this gives us little problem, since the multiple nearest cases may be ordered as equal for the user to select. However, if we are using k-NN with interpolation, we do not want to interpolate between cases which are not close in the input domain. Fig. 9 shows an inappropriate interpolation.



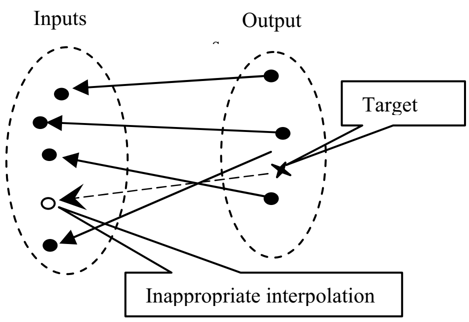


Fig. 8. Direct problem (single-valued solutions) **Fig. 9.** Inverse problem (multiple solutions)

4.2 Constraints

In fact, the particles degradation problem is seemingly a constraint satisfaction problem. The SQL example given above shows 2 types of constraints: continuous constraints of the type “Where *s-out*$_1$<10” and discrete constraints like “(*alpha = 30 OR alpha = 45 OR alpha = 50*)”. Constraints will involve both inputs and outputs.

One approach to dealing with constraints is to run the SQL query given above on the case base itself, beforehand, thus retrieving the nearest neighbour solution that satisfies the constraint. However, for sparse case-bases this may not be desirable, this may give poor solutions. Fig. 9 shows an example where there is a much more relevant case to a given target, which does not satisfy the constraint.

However, a variety of approaches were used in the past for handling constraints which might be a useful resource in this case. For instance, Borning *et al.* (1987) used constraint "hierarchies" to deal with the graphical display of a physical simulation. Hower (1989) implemented "sensitive relaxation" for resolving conflicts in the floor planning problem. Fox (1987) made use of the concept of "preferences" among relaxations in job-shop scheduling. Also, Freuder and Wallace (1992) proposed partial constraint satisfaction to overcome an over constrained problem so that a "good enough" solution can be found in the absence of a complete solution.

Also, Stanfill and Waltz (1986) used predictor restriction and goal restriction for restricting the database records by setting a dissimilarity threshold to retrieve records with smaller dissimilarity ratings. Portinale and Montani (2002) proposed a fuzzy case retrieval approach based on SQL. In their method, they extend the usual SQL query to a much wider use by setting a similarity threshold so that those cases having similarity degree greater than the specified threshold would be retrieved.

Another approach, currently under investigation by the authors is to add constraint variables to the target space, and re-define the target as a set of points in the extended domain. This method may have advantages in terms of interpolation for k-NN.

4.3 Definition of Metrics on the Query Space, an Interpolation over Nominal Values

The usability of a model is intimately connected with its queryability. The database model can be queried with SQL by selecting any set of variables we like to be inputs. If we need this to apply equally to our case-based model, then we need a similarity metric defined over the entire space of inputs and outputs. Often these inputs are nominal, and not ordered linearly. For example, in the pneumatic conveyor, there are several different types of bend that can be used (e.g., long radius, short radius elbow, turbulence drum, circular, box bend, blinded tee etc.). Similarities over these types are naturally defined by a similarity matrix. For k-NN methods, we shall need a method of interpolation over nominal values.

There are several approaches to nominal value interpolation. Stanfill and Waltz (1986) discussed several dissimilarity metrics used in prediction, such as the overlap metric, the weighted feature metric, and value difference metric. Chatterjee and Campbell (1993) treated nominal values as linearly ordered (i.e., with a restricted class of similarity metrics). In recent work, Knight and Woon (2003) proposed an algorithm called GSNN, which treats nominal values with a completely general metric. In their paper, they demonstrated, in a simple Iris example, the combined solution from both retrieved cases is distinct from the solution for both retrieved cases. Their approach was also tested on a discretised function with 21 discrete values and the test results shows that the GSNN algorithm can outperform conventional nearest neighbours methods such as k-NN and DWNN for evenly spread and randomly selected case bases.

4.4 The Experimental Planner

Experimental data is often collected by engineers in order to assist in design tasks. This data is often more detailed and reliable than modeled data. However, it is also sometimes much more expensive to produce. This is certainly the case with pneumatic conveyors. However, numerical models can be used to generate sparse databases, which may then be tested for accuracy of prediction. CBR reduction methods (Woon *et al.*, 2003; Salamo & Golobardes, 2002) can be used to reduce the number of cases in such a case base.

5 Conclusion and Future Work

In conclusion, we are now exploring the feasibility of using a case-based model as a tool to improve the usability of numerical models. This approach intends to exploit the advantage of multi-agent collaboration that makes use of the coordination of each agent's expertise. Such flexible interaction strategy allows iterative sharing of control between the user, the CBR system, and a numerical model to perform queries for a given target problem. In this paper, we have seen but the tip of the iceberg yet there may be advantages to be gained in using numerical model for adaptation validation and refining constraints. We are now looking at ways to implement such a case-based model and attempt to use other various reduction methods to build a case base for numerical use.

References

1. Agarwal, V. K., Mills, D., Mason, J.D., The Best of Bulk Solids Handling, Pneumatic Conveying of Bulk Powders, vol. D/86 Trans Tech Publications, Clausthal-Zellerfeld, Germany, (1985) 111-116
2. Bell, T. A., Boxman, A., Jacobs, J. B., Attrition of Salt during Pneumatic Conveying, Proceedings of The 5th World Congress of Chemical Engineering, San Diego, USA vol. V, (1996) 238-243.
3. Borning, A., Duisberg, R., Freeman-Benson, B., Kramer, A., Woolf, M., Constraint Hierarchies, Proceedings 1987 ACM Conference on Object-Oriented Programming Systems, Languages and Applications, Orlando, FL (1987) 48-60.
4. Bridge, D., Towards Conversational Recommender Systems: A Dialogue Grammar Approach, Workshop Proceedings of the 6th European Conference on Case-Based Reasoning, ECCBR-02, Aberdeen, Scotland, UK (2002).
5. Chatterjee, N., Campbell, J. A., Adaptation through Interpolation for Time Critical Case-Based Reasoning. Lecture Notes in Artificial Intelligence, Vol. 837: published by Springer-Verlag, 1st European Workshop, EWCBR-93, Kaiserslautern, Germany, November (1993) 221-233.
6. Cover, T. M., Hart, P., Nearest Neighbour Pattern Classification, IEEE Transactions on Information Theory, 13, (1967) 21-27.
7. Fox, M., Constraint Directed Search: A Case Study of Job-Shop Scheduling, Morgan Kaufmann, Los Altos, CA, 1987.

8. Freuder, E. C., Wallace, R. J., Partial Constraint Satisfaction, Artificial Intelligence 58 (1992) 21-70.
9. Hanson, R., Allsopp, D., Deng, T., Smith, D., Bradley, M. S. A., Hutchings, I. M., Patel, M. K., A Model to Predict the Life of Pneumatic Conveyor Bends, Proc Instn Mech Engrs Vol 216 Part E: J Process Mechanical Engineering, IMechE, (2002) 143-149.
10. Hilbert, J. D., The Best of Bulk Solids Handling, Pneumatic Conveying of Bulk Powders, vol. D/86 Trans Tech Publications, Clausthal-Zellerfeld, Germany, (1984) 107-110.
11. Hower, W., Sensitive Relaxation of an Overspecified Constraint Network, Proceedings Second International Symposium on Artificial Intelligence, Monterrey, Mexico (1989).
12. Kalman, H., Attrition of Powders and Granules at Various Bends during Pneumatic Conveying, Powder Technology 112, Elsevier Science S.A. (2000) 244-250.
13. Knight, B., Woon, F., Case Base Adaptation Using Solution-Space Metrics, to appear in: Proceedings of the 18th International Joint Conference on Artificial Intelligence, IJCAI-03, Acapulco, Mexico (2003).
14. Kolodner, J., Case-Based Reasoning, Morgan Kaufmann Publishers; ISBN: 1558602372; (November 1993).
15. Marcus, R. D., Hilbert, J. D., Klinzing, G. E., The Best of Bulk Solids Handling, Pneumatic Conveying of Bulk Powders, vol. D/86 Trans Tech Publications, Clausthal-Zellerfeld, Germany, (1985) 121-126.
16. Mitchell, T., Machine Learning, McGraw-Hill Series in Computer Science, WCB/McGraw-Hill, USA, (1997) 230 – 247.
17. Portinale, L., Montani, S., A Fuzzy Case Retrieval Approach Based on SQL, Lecture Notes in Artificial Intelligence, Vol. 2416: published by Springer-Verlag, 6th European Conference, ECCBR 2002, Aberdeen, Scotland, UK, September (2002) 321-335.
18. Salamo, M., Golobardes, E., Deleting and Building Sort Out Techniques for Case Base Maintenance, LNAI 2416: published by Springer, Germany.6th European Conference, ECCBR-02, September, Aberdeen, Scotland, UK (2002) 365-379.
19. Schach, S. R., Classical and Object-Oriented Software Engineering with UML and Java, McGraw-Hill International Editions, Computer Science Series (1999).
20. Stanfill, C., Waltz, D., Toward Memory-Based Reasoning, Communications of the ACM 29, 12, December, (1986) 1213-1228.
21. Weinberger, C. B., Shu, M. T., Helical Gas—Solids Flow II. Effect of Bend Radius and Solids Flow Rate on Transition Velocity, Powder Technology 48 (1986) 19-22.
22. Woon, F., Knight, B., Petridis, M., Case Base Reduction Using Solution-Space Metrics, to appear in: Proceedings of the 5th International Conference on Case-Based Reasoning, ICCBR-03, Trondheim, Norway (2003).